# Fixed points of $C^2$ maps

Layne T. Watson (*)


## ABSTRACT

S. N. Chow and J. A. Yorke have proposed in abstract terms an algorithm for computing fixed points of $C^2$ maps that is *globally convergent* with probability one. A numerical implementation of that algorithm is presented here, where careful attention has been paid to computational efficiency, accuracy, and robustness. Convergence proofs for the numerical algorithm require differential geometry, and are given elsewhere. FORTRAN subroutines are given and explained in detail, and some typical numerical results are presented. It is shown how to modify the subroutines to compute zeros and handle some large sparse problems.


## 1. INTRODUCTION

Homotopy methods (also known as continuation methods or methods of incremental loading) for computing zeros of nonlinear systems are well known [1,4,5,6,7,8,9,10,11,13]. In abstract terms a homotopy method is as follows : Let B be a Banach space and $f : B \to B$ the function whose zero is desired. Let $s : B \to B$ be a simple function with a known zero. Construct a continuous map (the homotopy) $\Phi : B \times [0,1] \to B$ such that $\Phi(x,0) = s(x)$ and $\Phi(x,1) = f(x)$. Then by solving the equation $\Phi(x,\lambda)=0$ in $B \times [0,1]$, one attempts to move from the known zero of $s(x)$ (at $\lambda=0$) to the unknown zero of $f(x)$ (at $\lambda=1$). There is a considerable amount of theory concerning when this procedure will work [11], but in general, moving from a zero of $s(x)$ to a zero of $f(x)$ may or may not be possible. Sometimes $\lambda$ is treated as an independent variable [4,11], and sometimes $\lambda$ is a dependent variable with arc length or some other parameter as the independent variable [7,9,10,13].

For the calculation of (Brouwer) fixed points, the supporting theory is much more satisfactory. Let B be the closed unit ball (or any compact, convex subset) in $E^n$, and let $f : B \to B$ be a $C^2$ map. S. N. Chow, J. Mallet-Paret, and J. Yorke [3] have proved the following powerful theorem :

Define $\rho_a : [0,1) \times B \to E^n$ by

$$\rho_a(\lambda,x) = \lambda(x - f(x)) + (1 - \lambda)(x - a)$$

where $f : B \to B$ is a $C^2$ map such that the Jacobian matrix of $x-f(x)$ is nonsingular at every fixed point of f. Then for almost all a in the interior of B, the set $\{(\lambda,x) \mid 0 \leq \lambda < 1, x \in B, \rho_a(\lambda,x) = 0\}$ of zeros of $\rho_a$ consists of

1) a finite number of closed loops (having finite length) in $(0,1) \times B$;

2) a finite number of arcs (having finite length) in $(0,1) \times B$ with endpoints in $\{1\} \times B$;

3) a curve of finite length starting at $(0,a)$ and ending at $(1,\bar{x})$, where $\bar{x} \in B$ is a fixed point of f.

The curves in 1), 2), and 3) are disjoint and continuously differentiable.

In other words, with probability one there is a zero curve of $\rho_a(\lambda,x)$ emanating from $(0,a)$ which reaches a fixed point $\bar{x}$ of f (at $\lambda = 1$). This zero curve is smooth and has finite length. Thus computing a fixed point of f merely amounts to tracking a (smooth) zero curve of $\rho_a(\lambda,x)$. Note that the assumptions on f ($C^2$ smoothness and non-singularity of the Jacobian matrix of $x-f(x)$ at the fixed point) are quite mild considering that global convergence is guaranteed with probability one.

## 2. ALGORITHM

A brief outline of how the zero curve is followed will be given here. Complete details of the algorithm and convergence proofs are in [13]. The basic idea, which has been used by several authors [7], [9], [10], is to parameterize the zero curve of $\rho_a(\lambda,x)$ by arc length and then solve an initial value problem. The zero curve $(\lambda(s), x(s))$ emanating from $(0,a)$ satisfies

$$(1)\quad \rho_a[\lambda(s), x(s)] = 0,\ \lambda(0) = 0,\ x(0) = a$$

and thus is given by the solution of the initial value problem

$$(2)\quad \frac{d}{ds}\rho_a[\lambda(s),x(s)] = 0,$$
$$\left\|\left(\frac{d\lambda}{ds}, \frac{dx}{ds}\right)\right\| = 1,\ \lambda(0) = 0,\ x(0) = a.$$

(*) L. T. Watson, Virginia Polytechnic Institute and State University, Dept. of Computer Science, Blacksburg, Va. 24061, USA.

Computing $(d\lambda/ds, dx/ds)$ reduces to finding the kernel of the $n \times (n+1)$ matrix, $[a - f(x), I - \lambda Df(x)]$ which always has full rank [3,13]. Therefore there are no "singular points" along the curve.
Of course the zero curve of $\rho_a(\lambda,x)$ could be followed by a general curve tracking program like that of Kubicek [10], but since the ultimate objective is a fixed point and not the zero curve of $\rho_a$, special tactics are called for. There are several other reasons for not using the program in [10]. That program produces a specified number of data points on the curve, and it would be difficult to make it stop exactly at $\lambda(s) = 1$. Furthermore, the numerical linear algebra and ODE techniques in that program are primitive compared to current technology (as, for example, [2] and [12]).

## 3. DETAILS ON THE FORTRAN SUBROUTINES

There are seven subroutines : FIXPT, FODE, DCPOSE, INNER, STEP, INTRP, and ROOT. The user need only call FIXPT; all the others are directly or indirectly used by FIXPT. The calling sequence is

CALL FIXPT (N, Y, ARCTOL, EPS, ARCLEN, NFE, IFLAG)

where : N is the dimension of x and f(x).
Y is a vector of length N+1, which contains the pair $(\bar\lambda, \bar x)$ on return. For a normal return, $\bar\lambda = 1$ and $\bar x$ is an approximate fixed point of f(x).
ARCTOL is the local error allowed the ODE solver when tracking the zero curve of $\rho_a(\lambda,x)$. FIXPT automatically decreases this local error allowance where the curvature is large, but ARCTOL is used whenever possible. If ARCTOL is too large, the algorithm will wander too far from the true zero curve, which is reported to the user by IFLAG = 5. The theory behind the detection of this wandering is in [13]. On a normal return ARCTOL = EPS.
EPS is the (absolute) accuracy desired in the computed fixed point. Normally EPS is much smaller than ARCTOL. Computational experience has shown that a good rule of thumb is ARCTOL = SQRT (EPS). EPS too small is reported to the user by IFLAG = 2.
ARCLEN is the arc length of the zero curve followed. On an error return ARCLEN may be incorrect.
NFE is the number of Jacobian evaluations. The number of function evaluations (of f) is usually NFE+1, but may be one or two larger. NFE is correct on an error return.
IFLAG is the error flag. FIXPT should be called with IFLAG = 0. IFLAG = 1 on a normal return.
IFLAG = 2 indicates EPS (or ARCTOL) is too small. ARCTOL and EPS have been changed by FIXPT to appropriate values. The solution can be continued by calling FIXPT again without changing any parameters.
IFLAG = 3 indicates that a very large amount of work is being expended, which means that either the problem is very difficult or (more likely) ARCTOL is too small. The solution can be continued by calling FIXPT again with IFLAG = 3. IFLAG = 4 indicates a rank deficient Jacobian, a fatal error. IFLAG = 5 indicates that the computed solution has wandered too far from the true zero curve. The problem should be restarted with IFLAG = 0 and a smaller ARCTOL and/or EPS. IFLAG = 6 indicates difficulty near the fixed point, probably due to the singularity of the Jacobian matrix of x - f(x) at the fixed point. The calculated fixed point may be inaccurate.
N is an argument, ARCTOL, EPS, and IFLAG are transients, and Y, ARCLEN, and NFE are results.
The subroutines STEP, INTRP, and ROOT are from [12]. STEP is used to solve the initial value problem, and INTRP and ROOT are used to calculate the vector x(s) corresponding to $\lambda(s) = 1$. FODE specifies the ordinary differential equation for STEP. FODE must determine the kernel of a matrix, and it uses DCPOSE and INNER for this purpose. DCPOSE was taken from [2] with minor modifications.
The use of labelled common could have been avoided by modifying STEP. Instead we chose to use STEP verbatum and pass other necessary information via labelled common. Thus the user needs only one version of STEP, and future improvements in STEP can be easily incorporated into this fixed point package. The difficulty arises because FIXPT has parameters which must be passed to FODE via STEP, and DCPOSE must return information back up through FODE and STEP to FIXPT.
FIXPT and FODE contain dimension statements which limit N to 100. STEP and ROOT contain machine dependent constants. No other modifications are required by the user. The user must supply two subroutines, F(X,V) and FJAC (X,V,K). Subroutine F evaluates f at X and returns the result in (the vector) V. Subroutine FJAC evaluates the Kth column of the Jacobian matrix of f at X and returns the result in V. FJAC may, of course, produce finite difference approximations to the Jacobian matrix. The effect of finite difference approximations on the overall efficiency and accuracy of the algorithm has not been explored so far.

## 4. MODIFICATIONS FOR COMPUTING ZEROS

FIXPT can be modified to compute zeros instead of fixed points, but then there is no theoretical guarantee of convergence. The homotopy map would be

$$\rho_a(\lambda,x) = \lambda f(x) + (1-\lambda)(x-a),$$

and there are some local convergence theorems for this homotopy map [13]. The array A in FIXPT is the vector a in the homotopy map $\rho_a(\lambda,x)$. Initially FIXPT uses A=0, but the user may want to take advantage of a priori knowledge of the zero $\bar x$. This can be done by putting the labelled common statement 1020 in the main program and defining A there. Statements 1430 through 1440 should be replaced by

Y(J) = A(J-1)

Replace statements 1660 through 1700 by

```
CALL F(Y(2), YP)
DO 70 JW = 1,N
AOLD = A(JW)
A(JW) = Y(1)* YP(JW)/(1.0-Y(1)) +Y(JW+1)
IF (ABS(A(JW) - AOLD) . GT .  .95) GO TO 40.
```

The above changes permit using an estimate of the zero for A, and do not require A to lie in the unit ball. The following modifications are related to the Jacobian matrix of the homotopy map.
Replace statement 2860 by

```
100 QR(J,1) = TZ(J) - Y(J+1) + A(J)
```

Replace statements 2910, 2920 by

```
110 QR (J,KP1) = Y(1) * TZ(J)
120 QR (K,KP1) = 1.0-Y(1) +QR (K, KP1)
```

The user could also make the program more foolproof by returning the error code IFLAG = 7 if $N \leq 0$, $EPS \leq 0$, or IFLAG is not 0,2,3.

## 5. MODIFICATIONS FOR LARGE SPARSE PROBLEMS

FIXPT was designed for low dimensional ($n \leq 100$) problems where the Jacobian matrix of f is dense. It is not difficult to modify FIXPT for a problem where n is very large but the Jacobian matrix of f is sparse. Write the Jacobian matrix of the homotopy map like

$$D\rho_a(x,\lambda) = [I - \lambda D f(x), a - f(x)].$$

Note that the order of the variables has been switched. The subroutine DCPOSE essentially reduces $D \rho_a (x,\lambda)$ to upper triangular form. If DF (x) is sparse, then $D \rho_a (x,\lambda)$ can be reduced to upper triangular form efficiently by, e.g., plane rotations. Probably DCPOSE should be tailored to the particular structure of Df(x). FODE computes the kernel of the upper triangular matrix produced by DCPOSE. This calculation in FODE would have to be changed, and again should probably be tailored to the particular form of Df(x). Thus FODE and DCPOSE would require major changes, but all the other subroutines would remain the same (except, of course, for the DIMENSION statements in FIXPT). It might also be desirable to reduce the storage requirements of STEP as explained in [12].

## 6. TESTING

FIXPT has been tested on several hundred problems of mixed polynomial, exponential, and trigonometric types, with n ranging from 2 to 100. Some problems, for example an exponential with a rapidly oscillating trigonometric exponent, give the method a great deal of difficulty. On the other hand, high degree ($\geq 10$) polynomials in 100 dimensions are handled easily. When used properly, the performance of FIXPT has been entirely satisfactory. The computed fixed points are usually accurate to within the tolerance EPS, and have never been in error by more than 10 EPS. Some typical computational results are in [13].

The results shown below are for Brown's function

$$f_1(x) = x_1 - \left[\prod_{i=1}^{n} x_i - 1\right],$$

$$f_j(x) = x_j - \left[\sum_{i=1}^{n} x_i + x_j - (n+1)\right], j = 2, \ldots, n$$

which was suggested by R. Saigal as a particularly difficult problem (because the Jacobian matrix is ill-conditioned).

| n | ND | arc length | CPU time (s) |
|---|---|---|---|
| 5 | 52 | 2.71 | .5 |
| 10 | 74 | 3.73 | 2.3 |
| 15 | 97 | 4.49 | 6.9 |
| 20 | 73 | 5.16 | 10.3 |
| 25 | 81 | 5.70 | 19.6 |
| 30 | 108 | 6.19 | 40.3 |
| 35 | 121 | 6.65 | 69.1 |
| 40 | 121 | 7.08 | 98.0 |
| 45 | 123 | 7.48 | 134.7 |
| 50 | 129 | 7.85 | 187.8 |

ND is the number of Jacobian evaluations, and the CPU time is execution time on a CDC 6500.
EPS = 1.E-8. ARCTOL = 1.OE-3 for N = 5,10,20,25 and ARCTOL = 1.OE-5 for N = 15,30,35,40,45,50. For this problem the zero curve does not turn back, although there are examples [13] where the zero curve turns back many times. Therefore the use of the parameter $\lambda$ as a *dependent* variable is crucial.

## 7. REFERENCES


1. BOGGS, P. : "The solution of nonlinear systems of equations by A-stable integration techniques", SIAM J. Numer. Anal., 8 (1971) 767-785.
2. BUSINGER, P. and GOLUB, G. H. : "Linear least squares solutions by Householder transformations", Numer. Math., 7 (1965) 269-276.
3. CHOW, S.N.; MALLET-PARET, J. and YORKE, J. A.: "Finding zeros of maps : homotopy methods that are constructive with probability one", Math. Comp. 32 (1978) 887-899.
4. DAHLQUIST, G.; BJORCK, A. and ANDERSON, N. : *Numerical methods*, Prentice-Hall, Englewood Cliffs, New Jersey, 1974.
5. EAVES, B. C. : "Homotopies for the computation of fixed points", Math. Programming, 3 (1972) 1-22.
6. EAVES, B. C. and SAIGAL, R. : "Homotopies for computation of fixed points on unbounded regions", Math. Programming, 3 (1972) 225-237.
7. KELLER, H. B. : "Numerical solution of bifurcation and nonlinear eigenvalue problems", in *Applications of bifurcation theory*, Academic Press, New York, 1977.
8. KELLOGG, R. B.; LI, T.Y. and YORKE, J. : "A constructive proof of the Brouwer fixed-point theorem and computational results", SIAM J. Numer. Anal., 13 (1976) 473-483.
9. KLOPFENSTEIN, R.W. : "Zeros of nonlinear functions", J. ACM, 8 (1961) 336-373.
10. KUBICEK, M. : "Dependence of solutions of nonlinear systems on a parameter", ACM-TOMS, 2 (1976) 98-107.

11. ORTEGA, J. M. and RHEINBOLDT, W. C. : *Iterative solution of nonlinear equations in several variables,* Academic Press, New York, 1970.

12. SHAMPINE, L. F. and GORDON, M. K. : *Computer solution of ordinary differential equations : the initial value problem,* W. H. Freeman, San Francisco, 1975.

13. WATSON, L. T. : "A globally convergent algorithm for computing fixed points of $C^2$ maps", Appl. Math. Comput., to appear.

```
      SUBROUTINE FIXPT(N,Y,ARCTOL,EPS,ARCLEN,NFE,IFLAG)                  10
C                                                                        20
C     SUBROUTINE  FIXPT  FINDS A FIXED POINT OF THE N-DIMENSIONAL        30
C     VECTOR FUNCTION F(X), WHICH IS ASSUMED TO BE A C2 MAP OF THE       40
C     UNIT BALL INTO ITSELF.  THE EQUATION  X = F(X)  IS SOLVED          50
C     BY FOLLOWING THE ZERO CURVE OF THE HOMOTOPY MAP                    60
C                                                                        70
C      LAMBDA*(X - F(X)) + (1 - LAMBDA)*(X - A)  ,                       80
C                                                                        90
C     STARTING FROM LAMBDA = 0, A = 0.  THE CURVE IS PARAMETERIZED       100
C     BY ARC LENGTH S, AND IS FOLLOWED BY SOLVING THE ORDINARY           110
C     DIFFERENTIAL EQUATION  D(HOMOTOPY MAP)/DS = 0  FOR                 120
C     Y(S) = (LAMBDA(S), X(S)).                                          130
C                                                                        140
C                                                                        150
C     THE USER MUST SUPPLY A SUBROUTINE  F(X,V)  WHICH EVALUATES         160
C     F(X)  AT X AND RETURNS THE VECTOR F(X) IN V, AND A SUBROUTINE      170
C     FJAC(X,V,K)  WHICH RETURNS IN V THE KTH COLUMN OF THE              180
C     JACOBIAN MATRIX OF F(X) EVALUATED AT X.  FIXPT  DIRECTLY OR        190
C     INDIRECTLY USES THE SUBROUTINES  STEP , INTRP , ROOT , FODE ,      200
C     F , FJAC , DCPOSE , AND THE FUNCTION  INNER .  FIXPT  AND          210
C     FODE  CONTAIN DIMENSION STATEMENTS WHICH LIMIT N TO 100.           220
C     STEP  AND  ROOT  CONTAIN MACHINE DEPENDENT CONSTANTS.  NO          230
C     OTHER MODIFICATIONS BY THE USER ARE REQUIRED.                      240
C                                                                        250
C                                                                        260
C     ON INPUT:                                                          270
C                                                                        280
C     N  IS THE DIMENSION OF X AND F(X).                                 290
C                                                                        300
C     Y  IS AN ARRRAY OF LENGTH  N + 1.                                  310
C                                                                        320
C     ARCTOL  IS THE LOCAL ERROR ALLOWED THE ODE SOLVER WHEN             330
C        FOLLOWING THE ZERO CURVE.  IF  ARCTOL .LE. 0.0  ON INPUT        340
C        IT IS RESET TO  .5*SQRT(EPS).  NORMALLY  ARCTOL  SHOULD         350
C        BE CONSIDERABLY LARGER THAN EPS.                                360
C                                                                        370
C     EPS  IS THE LOCAL ERROR ALLOWED THE ODE SOLVER WHEN VERY           380
C        NEAR THE FIXED POINT.  EPS  IS APPROXIMATELY THE ABSOLUTE       390
C        ERROR IN THE COMPUTED FIXED POINT.                              400
C                                                                        410
C     IFLAG  CAN BE 0, 2, OR 3.  IFLAG  SHOULD BE 0 ON THE FIRST         420
C        CALL TO  FIXPT.  IN CERTAIN SITUATIONS  IFLAG  IS SET TO        430
C        2 OR 3 BY  FIXPT, AND  FIXPT  CAN BE CALLED AGAIN WITHOUT       440
C        CHANGING  IFLAG.                                                450
C                                                                        460
C     Y, ARCTOL, EPS, ARCLEN, NFE, AND IFLAG SHOULD ALL BE               470
C     VARIABLES IN THE CALLING PROGRAM.                                  480
C                                                                        490
C                                                                        500
C     ON OUTPUT:                                                         510
C                                                                        520
C     N  IS UNCHANGED.                                                   530
C                                                                        540
C     Y(1) = LAMBDA, (Y(2),...,Y(N+1)) = X, AND Y IS AN APPROXIMATE      550
C        ZERO OF THE HOMOTOPY MAP.  NORMALLY LAMBDA = 1 AND X IS A       560
C        FIXED POINT OF F(X).  IN ABNORMAL SITUATIONS LAMBDA MAY         570
C        ONLY BE NEAR 1 AND X IS NEAR A FIXED POINT.                     580
C                                                                        590
C     ARCTOL = EPS AFTER A NORMAL RETURN (IFLAG = 1).                    600
C                                                                        610
C     EPS  IS UNCHANGED AFTER A NORMAL RETURN (IFLAG = 1).  IT IS        620
C        INCREASED TO AN APPROPRIATE VALUE ON THE RETURN IFLAG = 2.      630
C                                                                        640
C     ARCLEN  IS THE LENGTH OF THE PATH FOLLOWED.                        650
C                                                                        660
C     NFE  IS THE NUMBER OF FUNCTION EVALUATIONS (= NUMBER OF            670
C        JACOBIAN EVALUATIONS).                                          680
C                                                                        690
C     IFLAG =                                                            700
C       0   CAUSES  FIXPT  TO INITIALIZE EVERYTHING (USE ON FIRST CALL).710
C                                                                        720
C       1   NORMAL RETURN.                                              730
C                                                                        740
C       2   SPECIFIED ERROR TOLERANCE CANNOT BE MET.  EPS HAS BEEN      750
C           INCREASED TO A SUITABLE VALUE.  TO CONTINUE, JUST CALL       760
C           FIXPT  AGAIN WITHOUT CHANGING ANY PARAMETERS.                770
C                                                                        780
C       3   STEP  HAS BEEN CALLED 1000 TIMES.  TO CONTINUE, CALL        790
C           FIXPT  AGAIN WITHOUT CHANGING ANY PARAMETERS.                800
C                                                                        810
```

```
C        4    JACOBIAN MATRIX DOES NOT HAVE FULL RANK.  THE ALGORITHM      820
C             HAS FAILED (THE ZERO CURVE OF THE HOMOTOPY MAP CANNOT BE     830
C             FOLLOWED ANY FURTHER).                                       840
C                                                                          850
C        5    EPS  (OR  ARCTOL ) IS TOO LARGE.  THE PROBLEM SHOULD BE      860
C             RESTARTED BY CALLING  FIXPT  WITH A SMALLER  EPS  (OR        870
C             ARCTOL ) AND  IFLAG = 0.                                     880
C                                                                          890
C        6    I - DF(X)  IS NEARLY SINGULAR AT THE FIXED POINT.  ANSWER    900
C             MAY NOT BE ACCURATE.                                         910
C                                                                          920
C     *****  ARRAY DECLARATIONS.  *****                                   930
C                                                                          940
C     THE FOLLOWING ARRAYS ARE FOR A MAXIMUM OF  N = 100 VARIABLES.        950
C     TO HANDLE UP TO  N  VARIABLES, CHANGE EACH 100 AND 101 TO  N         960
C     AND  N + 1  RESPECTIVELY.                                            970
C                                                                          980
C     ARRAYS NEEDED BY THE ODE SUBROUTINE  STEP .                          990
      REAL Y(101),WT(101),PHI(101,16),P(101),YP(101),PSI(12)              1000
C                                                                         1010
      COMMON /FIXEDP/ YPOLD(101),A(100),NFEC,NC,NP1,IFLAGC                1020
C                                                                         1030
C     *****  END OF DIMENSIONAL INFORMATION.  *****                       1040
C                                                                         1050
      EXTERNAL FODE                                                       1060
      LOGICAL START,CRASH,ST99                                            1070
C                                                                         1080
C     LIMITD  IS AN UPPER BOUND ON THE NUMBER OF STEPS.  IT MAY BE        1090
C     CHANGED BY CHANGING THE FOLLOWING DATA STATEMENT:                   1100
      DATA LIMITD/1000/                                                   1110
C                                                                         1120
C                                                                         1130
C     :  :  :  :  :  :  :  :  :  :  :  :  :  :  :  :  :  :  :  :  :       1140
      IF (IFLAG .EQ. 0) GO TO 10                                          1150
      IF (IFLAG .EQ. 2) GO TO 35                                          1160
      IF (IFLAG .EQ. 3) GO TO 30                                          1170
C     ONLY VALID INPUT FOR  IFLAG  IS 0, 2, 3.                            1180
      RETURN                                                              1190
C                                                                         1200
C     *****  INITIALIZATION BLOCK.  *****                                 1210
C                                                                         1220
10    ARCLEN=0.0                                                          1230
      S=0.0                                                               1240
      IF (ARCTOL .LE. 0.0) ARCTOL=.5*SQRT(EPS)                            1250
      NFEC=0                                                              1260
      NC=N                                                                1270
      IFLAGC=0                                                            1280
      NP1=N+1                                                             1290
      SQNP1=SQRT(FLOAT(NP1))                                              1300
      CURTOL=.01/SQNP1                                                    1310
      ST99=.FALSE.                                                        1320
      START=.TRUE.                                                        1330
      CRASH=.FALSE.                                                       1340
      H=.1                                                                1350
      EPSSTP=ARCTOL                                                       1360
C     SET INITIAL CONDITIONS FOR ORDINARY DIFFERENTIAL EQUATION.          1370
      YPOLD(1)=1.0                                                        1380
      Y(1)=0.0                                                            1390
      WT(1)=1.0                                                           1400
      DO 20 J=2,NP1                                                       1410
        YPOLD(J)=0.0                                                      1420
        Y(J)=0.0                                                          1430
        A(J-1)=0.0                                                        1440
        WT(J)=1.0                                                         1450
20    CONTINUE                                                            1460
30    LIMIT=LIMITD                                                        1470
C                                                                         1480
C     *****  END OF INITIALIZATION BLOCK.  *****                          1490
C                                                                         1500
C                                                                         1510
C     *****  MAIN LOOP.  *****                                            1520
C                                                                         1530
35    DO 150 ITER=1,LIMIT                                                 1540
      IF (Y(1) .GE. 0.0) GO TO 50                                         1550
40    IFLAG=5                                                             1560
      RETURN                                                              1570
50    IF (S .LE. 7.0*SQNP1) GO TO 80                                      1580
C     ARC LENGTH IS GETTING TOO LONG, THE PROBLEM WILL BE                 1590
C     RESTARTED WITH A DIFFERENT  A  VECTOR.                              1600
      ARCLEN=ARCLEN+S                                                     1610
      S=0.0                                                               1620
```

```
60    START=.TRUE.                                                          1630
      CRASH=.FALSE.                                                         1640
C     COMPUTE A NEW  A  VECTOR.                                             1650
      CALL F(Y(2),A)                                                        1660
      DO 70 JW=1,N                                                          1670
        A(JW)=(Y(JW+1)-Y(1)*A(JW))/(1.0-Y(1))                               1680
C       THE NEW  A  VECTOR SHOULD BE WELL WITHIN THE UNIT BALL.             1690
        IF (ABS(A(JW)) .GT. .95) GO TO 40                                   1700
70    CONTINUE                                                              1710
      GO TO 100                                                             1720
80    IF (Y(1) .LE. .99  .OR. ST99) GO TO 100                               1730
C     WHEN LAMBDA REACHES .99, THE PROBLEM WILL BE RESTARTED WITH           1740
C     A NEW  A  VECTOR.                                                     1750
90    ST99=.TRUE.                                                           1760
      EPSSTP=EPS                                                            1770
      ARCTOL=EPS                                                            1780
      GO TO 60                                                              1790
C                                                                           1800
C     TAKE A STEP ALONG THE CURVE.                                          1810
100   CALL STEP(S,Y,FODE,NP1,H,EPSSTP,WT,START,HOLD,K,KOLD,CRASH,           1820
     +          PHI,P,YP,PSI)                                               1830
      NFE=NFEC                                                              1840
C     CHECK IF THE STEP WAS SUCCESSFUL.                                     1850
      IF (IFLAGC .NE. 4) GO TO 120                                          1860
      IFLAG=4                                                               1870
      RETURN                                                                1880
120   IF (.NOT. CRASH) GO TO 130                                            1890
C     RETURN CODE FOR ERROR TOLERANCE TOO SMALL.                            1900
      IFLAG=2                                                               1910
C     CHANGE ERROR TOLERANCES.                                              1920
      EPS=EPSSTP                                                            1930
      IF (ARCTOL .LT. EPS) ARCTOL=EPS                                       1940
C     CHANGE LIMIT ON NUMBER OF ITERATIONS.                                 1950
      LIMIT=LIMIT-ITER                                                      1960
      RETURN                                                                1970
C                                                                           1980
130   EPSSTP=ARCTOL                                                         1990
      IF (ABS(YP(1)) .LE. CURTOL) EPSSTP=EPS                                2000
      IF (Y(1) .LT. 1.0) GO TO 150                                          2010
      IF (ST99) GO TO 160                                                   2020
C                                                                           2030
C     IF LAMBDA .GE. 1.0 BUT THE PROBLEM HAS NOT BEEN RESTARTED             2040
C     WITH A NEW  A  VECTOR, BACK UP AND RESTART.                           2050
C                                                                           2060
      S99=S-.5*HOLD                                                         2070
C     GET AN APPROXIMATE ZERO Y(S) WITH  Y(1)=LAMBDA .LT. 1.0  .            2080
135   CALL INTRP(S,Y,S99,WT,P,NP1,KOLD,PHI,PSI)                             2090
      IF (WT(1) .LT. 1.0) GO TO 140                                         2100
      S99=.5*(S-HOLD+S99)                                                   2110
      GO TO 135                                                             2120
C                                                                           2130
140   DO 144 JUDY=1,NP1                                                     2140
        Y(JUDY)=WT(JUDY)                                                    2150
        YPOLD(JUDY)=P(JUDY)                                                 2160
144   WT(JUDY)=1.0                                                          2170
      S=S99                                                                 2180
      GO TO 90                                                              2190
C                                                                           2200
150   CONTINUE                                                              2210
C                                                                           2220
C     *****  END OF MAIN LOOP.  *****                                       2230
C                                                                           2240
C     LAMBDA HAS NOT REACHED 1 IN 1000 STEPS.                               2250
      IFLAG=3                                                               2260
      RETURN                                                                2270
C                                                                           2280
C                                                                           2290
C     USE INVERSE INTERPOLATION TO GET THE ANSWER AT LAMBDA = 1.0 .         2300
C                                                                           2310
160   SA=S-HOLD                                                             2320
      SB=S                                                                  2330
      LCODE=1                                                               2340
170   CALL ROOT(SOUT,Y1SOUT,SA,SB,EPS,EPS,LCODE)                            2350
C     ROOT  FINDS S SUCH THAT Y(1)(S) = LAMBDA = 1 .                        2360
      IF (LCODE .GT. 0) GO TO 190                                           2370
      CALL INTRP(S,Y,SOUT,WT,P,NP1,KOLD,PHI,PSI)                            2380
      Y1SOUT=WT(1)-1.0                                                      2390
      GO TO 170                                                             2400
190   IFLAG=1                                                               2410
C     SET IFLAG = 6 IF  ROOT  COULD NOT GET  LAMBDA = 1.0  .                2420
      IF (LCODE .GT. 2) IFLAG=6                                             2430
```

```
      ARCLEN=ARCLEN+SA                                                  2440
C     LAMBDA(SA) = 1.0 .                                                2450
      CALL INTRP(S,Y,SA,WT,P,NP1,KOLD,PHI,PSI)                          2460
C                                                                       2470
      DO 210 J=1,NP1                                                    2480
210   Y(J)=WT(J)                                                        2490
      RETURN                                                            2500
      END                                                               2510
      SUBROUTINE FODE(S,Y,YP)                                           2520
C                                                                       2530
CCC   SUBROUTINE  FODE  IS USED BY SUBROUTINE  STEP  TO SPECIFY THE     2540
CCC   ORDINARY DIFFERENTIAL EQUATION  DY/DS = G(S,Y) , WHOSE SOLUTION   2550
CCC   IS THE ZERO CURVE OF THE HOMOTOPY MAP.  S = ARC LENGTH,           2560
CCC   YP = DY/DS, AND  Y(S) = (LAMBDA(S), X(S)) .                       2570
CCC                                                                     2580
CCC   *****  ARRAY DECLARATIONS.  *****                                2590
CCC                                                                     2600
CCC   THE FOLLOWING ARRAYS ARE FOR A MAXIMUM OF N = 100 VARIABLES.      2610
CCC   TO HANDLE UP TO  N  VARIABLES, CHANGE EACH 100 AND 101 TO  N      2620
CCC   AND  N + 1  RESPECTIVELY.                                         2630
C                                                                       2640
      REAL Y(101),YP(101)                                               2650
C     ARRAYS FOR COMPUTING THE JACOBIAN MATRIX AND ITS KERNAL.          2660
      REAL QR(100,101),ALPHA(100),TZ(101)                               2670
      INTEGER PIVOT(101)                                                2680
C                                                                       2690
      REAL INNER                                                        2700
      COMMON /FIXEDP/ YPOLD(101),A(100),NFE,N,NP1,IFLAG                 2710
      NDIM=100                                                          2720
CCC                                                                     2730
CCC   *****  END OF DIMENSIONAL INFORMATION.  *****                     2740
C                                                                       2750
      NFE=NFE+1                                                         2760
C     NFE CONTAINS THE NUMBER OF JACOBIAN EVALUATIONS.                  2770
CCC * *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *     2780
CCC                                                                     2790
CCC   COMPUTE THE JACOBIAN MATRIX, STORE IT IN QR.                      2800
CCC                                                                     2810
CCC        QR = ( A - F(X), I - LAMBDA*DF(X) )  .                       2820
C                                                                       2830
      CALL F(Y(2),TZ)                                                   2840
      DO 100 J=1,N                                                      2850
100   QR(J,1)=A(J)-TZ(J)                                                2860
      DO 120 K=1,N                                                      2870
        CALL FJAC(Y(2),TZ,K)                                            2880
        KP1=K+1                                                         2890
        DO 110 J=1,N                                                    2900
110     QR(J,KP1)=-Y(1)*TZ(J)                                           2910
120   QR(K,KP1)=1.0+QR(K,KP1)                                           2920
C                                                                       2930
CC  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *     2940
C     REDUCE THE JACOBIAN MATRIX TO UPPER TRIANGULAR FORM.              2950
      CALL DCPOSE(NDIM,N,QR,ALPHA,PIVOT,IERR,TZ,YP)                     2960
      IF (IERR .EQ. 0) GO TO 20                                         2970
      IFLAG=4                                                           2980
      RETURN                                                            2990
C     COMPUTE KERNAL OF JACOBIAN, WHICH SPECIFIES YP=DY/DS.             3000
20    TZ(NP1)=1.0                                                       3010
      DO 40 LW=1,N                                                      3020
        I=NP1-LW                                                        3030
        IK=I+1                                                          3040
        SUM=0.0                                                         3050
        DO 30 J=IK,NP1                                                  3060
30      SUM=SUM+QR(I,J)*TZ(J)                                           3070
40    TZ(I)=-SUM/ALPHA(I)                                               3080
      YPNORM=SQRT(INNER(1,NP1,TZ,TZ,0.0))                               3090
      DO 60 K=1,NP1                                                     3100
60    YP(PIVOT(K))=TZ(K)/YPNORM                                         3110
      IF (INNER(1,NP1,YP,YPOLD,0.0) .GE. 0.0) GO TO 80                  3120
      DO 70 I=1,NP1                                                     3130
70    YP(I)=-YP(I)                                                      3140
CC                                                                      3150
CC    SAVE CURRENT DERIVATIVE (= TANGENT VECTOR) IN  YPOLD .            3160
80    DO 90 I=1,NP1                                                     3170
90    YPOLD(I)=YP(I)                                                    3180
      RETURN                                                            3190
      END                                                               3200
      REAL FUNCTION INNER(K,M,A,B,C)                                    3210
      DIMENSION A(1),B(1)                                               3220
      SUM=C                                                             3230
      DO 10 I=K,M                                                       3240
```

```
10      SUM=SUM+A(I)*B(I)                                                  3250
        INNER=SUM                                                          3260
        RETURN                                                             3270
        END                                                                3280
        SUBROUTINE DCPOSE(NDIM,N,QR,ALPHA,PIVOT,IERR,Y,SUM)               3290
C                                                                          3300
CC      SUBROUTINE  DCPOSE  IS A MODIFICATION OF THE ALGOL PROCEDURE       3310
CC      DECOMPOSE  IN P. BUSINGER AND G. H. GOLUB, LINEAR LEAST            3320
CC      SQUARES SOLUTIONS BY HOUSEHOLDER TRANSFORMATIONS,                  3330
CC      NUMER. MATH. 7 (1965) 269-276.                                     3340
C                                                                          3350
        INTEGER NDIM,N,PIVOT(1)                                            3360
        REAL QR(NDIM,1),ALPHA(N)                                           3370
        INTEGER IERR,I,J,JBAR,K                                            3380
        REAL BETA,SIGMA,ALPHAK,QRKK,Y(1),SUM(1)                            3390
        REAL INNER                                                         3400
        IERR=0                                                             3410
        NP1=N+1                                                            3420
        DO 20 J=1,NP1                                                      3430
          SUM(J)=INNER(1,N,QR(1,J),QR(1,J),0.0)                            3440
20      PIVOT(J)=J                                                         3450
        DO 500 K=1,N                                                       3460
          SIGMA=SUM(K)                                                     3470
          JBAR=K                                                           3480
          KP1=K+1                                                          3490
          DO 40 J=KP1,NP1                                                  3500
            IF (SIGMA .GE. SUM(J)) GO TO 40                                3510
            SIGMA=SUM(J)                                                   3520
            JBAR=J                                                         3530
40        CONTINUE                                                         3540
          IF (JBAR .EQ. K) GO TO 70                                        3550
          I=PIVOT(K)                                                       3560
          PIVOT(K)=PIVOT(JBAR)                                             3570
          PIVOT(JBAR)=I                                                    3580
          SUM(JBAR)=SUM(K)                                                 3590
          SUM(K)=SIGMA                                                     3600
          DO 50 I=1,N                                                      3610
            SIGMA=QR(I,K)                                                  3620
            QR(I,K)=QR(I,JBAR)                                             3630
            QR(I,JBAR)=SIGMA                                               3640
50        CONTINUE                                                         3650
C         END OF COLUMN INTERCHANGE.                                       3660
70        SIGMA=INNER(K,N,QR(1,K),QR(1,K),0.0)                             3670
          IF (SIGMA .NE. 0.0) GO TO 60                                     3680
          IERR=1                                                           3690
          RETURN                                                           3700
60        IF (K .EQ. N) GO TO 500                                          3710
          QRKK=QR(K,K)                                                     3720
          ALPHAK=-SQRT(SIGMA)                                              3730
          IF (QRKK .LT. 0.0) ALPHAK=-ALPHAK                                3740
          ALPHA(K)=ALPHAK                                                  3750
          BETA=1.0/(SIGMA-QRKK*ALPHAK)                                     3760
          QR(K,K)=QRKK-ALPHAK                                              3770
          DO 80 J=KP1,NP1                                                  3780
80        Y(J)=BETA*INNER(K,N,QR(1,K),QR(1,J),0.0)                         3790
          DO 100 J=KP1,NP1                                                 3800
            DO 90 I=K,N                                                    3810
              QR(I,J)=QR(I,J)-QR(I,K)*Y(J)                                 3820
90          CONTINUE                                                       3830
            SUM(J)=SUM(J)-QR(K,J)**2                                       3840
100       CONTINUE                                                         3850
500     CONTINUE                                                           3860
        ALPHA(N)=QR(N,N)                                                   3870
        RETURN                                                             3880
        END                                                                3890
```